Qcluster: Relevance Feedback Using Adaptive Clustering for Content-Based Image Retrieval

Deok-Hwan Kim
Division of Information and Communication Engineering
Korea Advanced Institute of Science and Technology
373-1, Gusong-dong, Yusong-gu, Daejeon 305-701, South Korea
dhkim@islab.kaist.ac.kr

Chin-Wan Chung
Division of Computer Science
Korea Advanced Institute of Science and Technology
373-1, Gusong-dong, Yusong-gu, Daejeon 305-701, South Korea
chungcw@islab.kaist.ac.kr

ABSTRACT

The learning-enhanced relevance feedback has been one of the most active research areas in content-based image retrieval in recent years. However, few methods using the relevance feedback are currently available to process relatively complex queries on large image databases. In the case of complex image queries, the feature space and the distance function of the user's perception are usually different from those of the system. This difference leads to the representation of a query with multiple clusters (i.e., regions) in the feature space. Therefore, it is necessary to handle disjunctive queries in the feature space.

In this paper, we propose a new content-based image retrieval method using adaptive classification and cluster-merging to find multiple clusters of a complex image query. When the measures of a retrieval method are invariant under linear transformations, the method can achieve the same retrieval quality regardless of the shapes of clusters of a query. Our method achieves the same high retrieval quality regardless of the shapes of clusters of a query since it uses such measures. Extensive experiments show that the result of our method converges to the user's true information need fast, and the retrieval quality of our method is about 22% in recall and 20% in precision better than that of the query expansion approach, and about 34% in recall and about 33% in precision better than that of the query point movement approach, in MARS.

Keywords

relevance feedback, image database, classification, cluster-merging, content-based image retrieval

1. INTRODUCTION

With the advances in image processing, information retrieval, and database management, there have been extensive studies on content-based image retrieval (CBIR) for large image databases. Content-based image retrieval systems use the visual contents of images, such as color, texture, and shape features to represent and index images.

Many retrieval methods represent images as vectors in the feature space and many retrieval systems take a query image, features and feature representations, on which the search is based on, as the input from a user. A range query or a nearest-neighbor query are used to retrieve images similar to a query image in terms of the features and feature representations provided by a user. That is, the closer two vectors are, the more similar the corresponding images are.

However, it is not an easy task for a user to select features and feature representations which are effective to express high-level concepts (i.e., objects) in a query image. The reasons are as follows. First, there exists a gap between high-level concepts and low-level feature representations [16]. So, the current technology requires interactive user help to map from low-level feature representations to high-level concepts. Second, the user's preference for a query image changes from time to time since he or she may not initially have the query image at hand or an initial query may evolve to an ideal query during the retrieval process.

To resolve this problem, recent studies in CBIR have focused on the approach based on the relevance feedback. The relevance feedback is an automatic refining process of the current query to representations based on low-level features using the user's evaluation of the relevance of images retrieved by query processing [16]. Whenever the system presents a set of images considered to be similar to a given query, the user can pick up the ones the most relevant to the given query and the system refines the query using them. Let the relevant images be the ones picked up by the user.

Implementing the relevance feedback concerns the computation of a new query point (or points) in a feature space and the change of a distance function. As shown in Figure 1(a), early studies [11, 15] represent a new query as a single point and change the weights of feature components to find an optimal query point and an optimal distance function. In this case, a single point is computed using the weighted average of all relevant images in the feature space. The contours represent equi-similarity lines. Meanwhile, a recent study

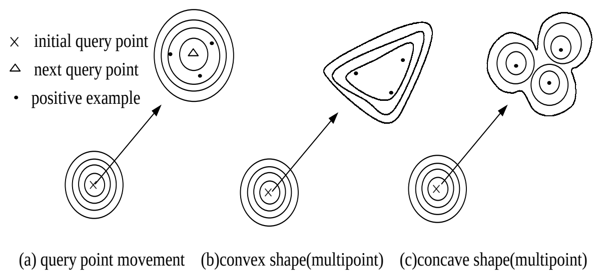

Figure 1: Query shape

[13] represents a new query as multiple points to determine the shape of the contour as shown in Figure 1(b). This approach uses a clustering method [8] to compute new query points using query results (relevant images) based on the user's relevance judgement. It is assumed that the relevant images are mapped to points close together according to the similarity measure. A single large contour is constructed to cover all query points and the system finds images similar to them. However, if the feature space and the distance function of the user's perception are quite different from those of the system, the relevant images are mapped to disjoint clusters of arbitrary shapes in the feature space. That is, the relevant images may be ranked below other retrieved images for the given query. In order to converge rapidly to the user's information need, the system should find the images similar to any of the query points as in Figure 1(c). A query that retrieves the images similar to any of the query points is called a disjunctive query. Especially, a complex image query is represented as disjoint multiple clusters.

In this paper, we propose a new adaptive classification and cluster-merging method to determine arbitrary shapes of contours for a given complex image query. Also we propose an approach to the relevance feedback using multiple query points to support disjunctive queries.

1.1 Brief Sketch of Our Method

Figure 2 shows the proposed relevance feedback mechanism. At the first stage, an example image submitted by the user is parsed to generate an initial query $Q = (q, d, k)$, where q is a query point in the feature space, k is the number of images in the query result to be returned by the system, and d is the distance function. The query point q is compared with images in the database using the distance function d. According to d, the result set consisting of k images close to q, $Result(Q) = \{p_1, \ldots, p_k\}$, is returned to the user.

At the next stage, the user evaluates the relevance of images in $Result(Q)$ by assigning a relevance score to each of them. Based on those scores, the relevant set, $Relevant(Q) = \{p'_1, \ldots, p'_m\}$, is obtained. In this paper, we present a new adaptive clustering method consisting of two processes: the classifying process and the cluster-merging process. The proposed classifying process places each element of the relevant set, $Relevant(Q)$, in one of the current clusters or a new cluster. Then, the proposed cluster-merging process reduces the number of clusters by merging certain clusters to reduce the number of query points in the next iteration. Finally, representatives of clusters generated from relevant images in the classified set make up the set of new query points. A new query, $Q' = (q', d', k)$ with a set of new query points q' and a new distance function d', is computed and then used as an input for the second round.

After some iterations, the loop ends up with the final result set close to $Result(Q_{opt})$, where $Q_{opt} = (q_{opt}, d_{opt}, k)$ is the optimal query.

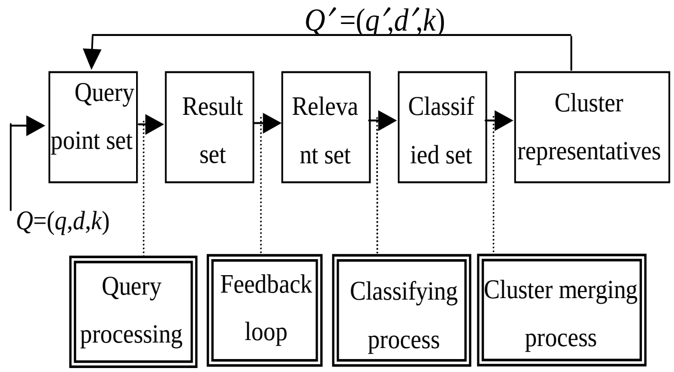

Figure 2: Overall structure of the proposed method

Our approach to the relevance feedback allows multiple objects to be a query. We refer to them as a multipoint query. When the user marks several points as relevant, we cluster sets of relevant points and choose the centroids of the clusters as their representatives. Then, we construct a multipoint query using a small number of good representative points. At the classifying process, a Bayesian classification function [9] is used. Statistics such as mean and covariance of each cluster, which were computed from the previous iteration, are used as the prior information. At the cluster-merging process, Hotelling's T^2 [12] is used to merge any pair of clusters in arbitrary shapes.

1.2 Contributions

The contributions of this paper are as follows:

- The adaptive clustering generates contours consisting of multiple hyper-ellipsoids, and therefore our retrieval method can handle disjunctive queries.

- Our method constructs clusters and changes them without performing complete re-clustering. Its computing time is short since the same statistical measures are used at both the classification stage and the cluster-merging stage.

- The measures used in our method are invariant under linear transformations. Therefore, the retrieval quality is the same regardless of the shapes of clusters of a query.

- Our experimental results show that the proposed method achieves about 22% improvement of recall and 20% improvement of precision against the query expansion approach[13], and about 35% improvement of recall and about 31% improvement of precision against the query point movement approach[15], in MARS.

1.3 Paper Organization

Section 2 provides a survey of related works with a brief discussion on the relevance feedback. Section 3 includes some interesting motivating examples, the similarity measure, and the overall algorithm of the multipoint relevance feedback. The classification and cluster-merging processes are described in Section 4 with an algorithm to select an appropriate group. Extensive experiments on a large set of 30,000 heterogeneous images and their results are reported in Section 5.

2. RELATED WORK

Earlier approaches [10, 18] to the content-based multimedia retrieval do not adapt the query and retrieval model based on the user's perception of the visual similarity. To overcome this problem, a number of relevance feedback techniques [1, 2, 4, 5, 11, 13, 15, 16, 19, 20, 21] have been proposed. They try to establish the link between high-level concepts and low-level feature representations and model the user's subjective perception from the user's feedback. There are two components to learn in the relevance feedback: a distance function and a new query point. The distance function is changed by learning weights of feature components, and the new query point is obtained by learning the ideal point that the user looks for.

The query-point movement has been applied to the image retrieval systems such as MARS [15] and MindReader [11]. These systems represent the query as a single point in the feature space and try to move this point toward "good" matches, as well as to move it away from "bad" result points. This idea originated from the Rochio's formula [14], which has been successfully used in document retrieval. In this approach, the re-weighting technique assigns a weight to each dimension of the query point. The weight is inversely proportional to the variance of feature values of the relevant points along that dimension. MARS uses a weighted Euclidean distance, which handles ellipsoids whose major axis is aligned with the coordinate axis. On the other hand, MindReader uses a generalized Euclidean distance, which permits the rotation of the axes so that it works well for arbitrarily oriented ellipsoids.

Recently, other query refinement methods using the multipoint relevance feedback were introduced. The query expansion approach [13] of MARS constructs local clusters for relevant points. In this approach, all local clusters are merged to form a single large contour that covers all query points. On the other hand, the query-point movement approach [11, 15] ignores these clusters and compute a single query point from all relevant points. These two approaches can generate a single hyper-ellipsoid or convex shapes using local clusters in some feature space to cover all query points for simple queries. However, both approaches fail to identify appropriate regions for complex queries. Wu et al. presented an aggregate dissimilarity model in FALCON [20], to facilitate learning disjunctive queries in the vector space as well as in arbitrary metric spaces. However, the proposed aggregate dissimilarity model depends on ad hoc heuristics and this model assumes that all relevant points are query points.

3. MULTIPOINT RELEVANCE FEEDBACK APPROACH

This section presents the overall mechanism of our approach to the multipoint relevance feedback. Table 1 shows some notations to be used.

Table 1: Symbols and their definitions

Symbol	Definition
p	dimension of feature vector
$x_{ij} = [x_{ij1}, \dots, x_{ijp}]'$	feature vector of jth image of ith cluster
$C_1, \dots, C_g$	g clusters
$\bar{x}_i = [\bar{x}_{i1}, \dots, \bar{x}_{ip}]'$	weighted centroid of ith cluster
$Q = \{\bar{x}_1, \dots, \bar{x}_g\}$	query set of g multiple points
$d^2()$	generalized Euclidean distance function
q_{opt}	ideal query point
n_i	the number of images for ith cluster
m_i	the sum of relevance score values of ith cluster
w_i	normalized weight of ith cluster
α	significance level
S_{pooled}^{-1}	pooled inverse covariance matrix
S_i^{-1}	inverse covariance matrix of ith cluster
v_{ij}	relevance score value for jth image of ith cluster
$\hat{d}()$	classifier function
$T^2()$	cluster-merging measure

3.1 Motivating Examples

EXAMPLE 1. *The user wants to select bird images via query-by-example in the image data set of 30,000 color images. A pairwise distance metric relying primarily on color is used to compare images. As shown in Figure 3, the set of retrieved relevant images includes bird images with a light-green background and ones with a dark-blue background. However, these may not be projected to points close together in the feature space. Instead, the points form two distinct clusters.* □

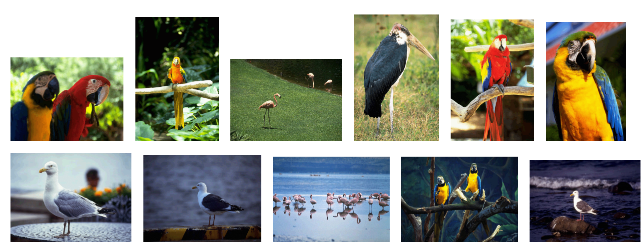

Figure 3: Bird images

Finding similar images in this space is related to clustering. If the difference between the user's perception and the feature representation in the system gets large, there comes a necessity of expressing a query by several points in the feature space. MARS uses multiple point queries and every query point is supposed to be merged. All relevant images are merged to several clusters and a single large contour is made to cover all representatives of these clusters.

EXAMPLE 2. *Given the top-leftmost image as a query in Figure 3, Figure 4 shows the 3 dimensional plot of feature*

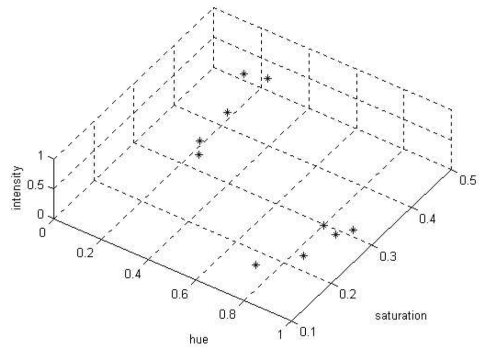

Figure 4: 3 dimensional plot of 10 points

vectors of 10 retrieved relevant images. It shows that five images are similar, but the other five images are quite different, MARS makes a single contour for two clusters. However, it is better to make separate contours for two different clusters of data. Our method can determine the shapes of two local clusters. □

For query expansion, Porkaew et al. [13] assumed that query images given by a user should be similar. Their method makes several clusters to include all relevant images and builds a large contour to cover them. So the single large contour of the clusters of a query is used as the basis for the search. However, the clusters of a query might be very apart from each other. So, the search might not yield fruitful results. For a successful search, the contours must be separated instead of being combined, as our method does.

A complex image query must be expressed as multiple query points so that multiple representatives of clusters are used.

The basic method of clustering image feature vectors is as follows: Initially, assign n input points (feature vectors of images) to n distinct clusters. Among all clusters, pick up the two clusters with the smallest distance between them. Merge them to form a new cluster. Repeat these two steps until there are no clusters left to be merged. We adjust the number of clusters using the Hotelling's T^2 and the significance level α. The basic idea of our approach is to use an adaptive classification and cluster merging method that finds multiple clusters of a query. Our goal is as follows:

- Given: user-selected p-dimensional points from the result of a k-nearest neighbor query and their relevance scores.

- Goal: find a set of centroids $\bar{x}_1, \ldots, \bar{x}_g$ of clusters and their weights $w_1, \ldots, w_g$.

We use multiple points, which are the centroids of clusters $C_1, \ldots, C_g$ to guess the best query point q_{opt}, their covariance matrices, and weights to learn the hidden distance function.

3.2 Similarity Measure

When a user marks several images as relevant ones at each iteration of the relevance feedback, we cluster a set of relevant points and choose the centroid of the cluster as its representative. For each image x in the database, the distance $d^2(x, \bar{x}_i)$ between the two points x and the centroid of the ith cluster $\bar{x}_i$ is defined by:

$$d^2(x, \bar{x}_i) = (x - \bar{x}_i)' S_i^{-1} (x - \bar{x}_i) \tag{1}$$

This quadratic distance function allows different weight for each dimension and it can express a user's high-level concept better than ordinary Euclidean distance, since it is an ellipsoid and an ellipsoid can express a user's hidden distance function better than a circle. MindReader [11] proved that this method is theoretically solid to handle similarity queries. The estimates $\bar{x}_i$ and S_i are as follows:

DEFINITION 1. *For n_i points of the ith cluster, the mean vector weighted by the relevance score v_{ik} is defined by:*

$$\bar{x}_i = \frac{\sum_{k=1}^{n_i} v_{ik} x_{ik}}{\sum_{k=1}^{n_i} v_{ik}} \tag{2}$$

DEFINITION 2. *The weighted covariance matrix S_i ($p \times p$ dimensional matrix) of the ith cluster is defined by:*

$$S_i = \sum_{k=1}^{n_i} v_{ik} (x_{ik} - \bar{x}_i)(x_{ik} - \bar{x}_i)' \tag{3}$$

The parameter S_i^{-1} can be estimated using an inverse matrix scheme(MindReader [11]) or a diagonal matrix scheme(MARS [13, 15]). Many relevance feedback methods make use of the sample covariance matrix S_i and its inverse [11, 16, 19, 21]. However, the singularity issue arises when the number of relevant images is smaller than the dimensionality of the feature space. In this case, regularization terms should be added on the diagonal of the covariance matrix before the inversion [21]. In this paper, we use a simpler form of d^2 using a diagonal matrix for S_i^{-1} instead of an inverse matrix.

Conventionally, a similarity query is represented as a single point, while we insist that a complex image query be represented as multiple points. We compute multiple representatives or a single representative using the proposed adaptive classification and cluster-merging method. A general aggregate distance function between a point x and a set of multiple query points $Q = \{\bar{x}_1, \ldots, \bar{x}_g\}$ is defined by [17, 20]:

$$d^{\alpha}_{aggregate}(Q, x) = \frac{1}{g} \sum_{i=1}^{g} d^{\alpha}(\bar{x}_i, x) \tag{4}$$

The negative value of α mimics a fuzzy OR function since the smallest distance will have the largest impact on the aggregate distance function [17]. For Equation (4), we use $\alpha = -2$ and incorporate the sum of relevance score values for each cluster. Consequently, we apply the following aggregate distance function to those representatives to find images similar to one of the representatives in the query point set.

$$d^2_{disjunctive}(Q, x) = \frac{\sum_{i=1}^{g} m_i}{\sum_{i=1}^{g} m_i / [(x - \bar{x}_i)' S_i^{-1} (x - \bar{x}_i)]} \tag{5}$$

where Q is a set of multiple cluster representatives $\{\bar{x}_1, \ldots, \bar{x}_g\}$, x is the feature vector of a target image, and m_i is the sum of relevance score values of ith cluster. This distance function can reflect disjunctive contours as shown in Example 3.

EXAMPLE 3. *The synthetic data consists of 10,000 points in $\Re^3$, randomly distributed uniformly within the axis-aligned* *cube $(-2,-2,-2) \sim (2,2,2)$. We used the aggregate distance function (Equation (5)) as the distance metric. In this case, S_i^{-1} is computed using a diagonal matrix scheme and m_i is set to 1 for all i. Points were retrieved if and only if they were within 1.0 units of either $(-1,-1,-1)$ or $(1,1,1)$. 820 points were retrieved. Figure 5 shows that this aggregate distance function can handle disjunctive queries.* □

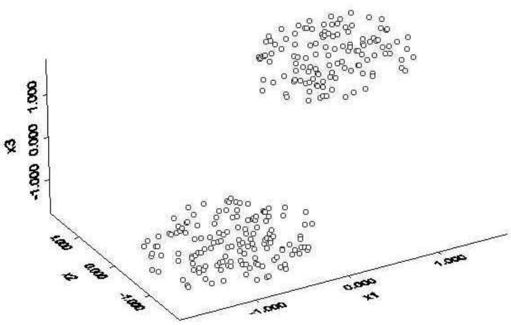

Figure 5: scatter plot for the result of the disjunctive query

3.3 A General Algorithm

We propose a novel relevance feedback approach for multipoint queries using the adaptive classification and cluster-merging method. The algorithm of our relevance feedback for multipoint queries is as follows:

Algorithm 1 k-nearest neighbor search
input: a query example
output: k retrieved images
begin
1. At the initial iteration, perform k-NN query. The user marks relevant images among ones retrieved for a given query and the relevant images are clustered using a hierarchical clustering method. **For** each current cluster, calculate a centroid, its covariance, and its weight.
2. Process k-NN query using the aggregate distance function to multiple representatives.
3. Continue or stop the query iteration.
4. Relevance feedback to the query results
5. **For** a new point in the relevant set,
6. Determine an appropriate cluster using an adaptive classification method
7. **If** a new point is located within the determined cluster boundary,
8. place it in the determined cluster
9. **Else** make it a separate cluster
10. **Endfor**
11. **For** each current cluster,
12. calculate a centroid, its covariance, and its weight.
13. **For** any pair of clusters,
14. **If** two clusters are close, then merge them.
15. **Endfor**
16. **Goto** Step 2 with the adjusted Q, and start a new iteration of retrieval.

end

4. ADAPTIVE CLASSIFICATION AND MERGING CLUSTERS

The adaptive classification and merging clusters are the cores of our approach. They are used to accelerate query processing by considering only a small number of representatives of the clusters, rather than the entire set of relevant images. When all relevant images are included in a single cluster, it is the same as MindReader's. At each stage the clusters are modified according to the result of a query and a user's relevance feedback. Therefore, it is necessary to construct new clusters without complete re-clustering.

The proposed method is composed of two stages: the classification stage and cluster-merging stage. At the first stage, new points are classified into current clusters or new clusters using their statistical information. At the second stage, the number of current clusters is reduced. Its advantages are as follows:

- It is easy to compute multiple query points since the method does not re-cluster completely at each iteration.
- The method can approximate any query shape to an arbitrarily oriented ellipsoid since the distance function is a quadratic form.
- The method can adjust the number of clusters using a rigorous statistical measure.
- The method using a diagonal matrix avoids the singularity problem and its performance is similar to that of the method using an inverse matrix.

4.1 Initial Clustering and Effective Radius

Initial clusters of the training data form the basis. Among numerous methods, we use the hierarchical clustering algorithm that groups data into hyperspherical regions. Once initial clusters are obtained, we calculate a mean vector $\bar{x}$, a weighted covariance matrix S, and an effective radius r. The mean vector determines the location of the hyperellipsoid, while the covariance matrix characterizes its shape and orientation. The weight of each cluster compared with the others is determined by the sum of relevance score values of points in each cluster. The effective radius is a critical value to decide whether a new point x lies inside the given ellipsoid.

LEMMA 1. *If x lies inside the ellipsoid, the following property is satisfied [3]:*

$$(x-\bar{x})'S^{-1}(x-\bar{x}) < r \qquad (6)$$

Let us assume that the data follows a Gaussian distribution and takes α as a significance level. For the given significance level α, $100(1-\alpha)\%$ (typically $95\% \sim 99\%$) of the data will fall inside the ellipsoid and the distance function follows a χ_p^2 distribution with p degrees of freedom. Then the effective radius r is $\chi_p^2(\alpha)$. As α decreases, a given effective radius increases. Any point outside of the ellipsoid is identified as an outlier and forms a new cluster.

4.2 Classification Stage

4.2.1 Bayesian Classifier with several clusters

Let $C_1,\ldots,C_g$ be g clusters. The classification algorithm places a new point in one of the g clusters or in a new cluster. A generalized Euclidean distance $D_i^2(x)$ between a new point x and the centroid of the ith cluster C_i, $\bar{x}_i$, is computed as follows:

$$D_i^2(x) = (x-\bar{x}_i)'S_{pooled}^{-1}(x-\bar{x}_i) \qquad (7)$$

where

$$S_{pooled} = \frac{1}{m_1+m_2+\cdots+m_g-g}[(m_1-1)S_1+(m_2-1)S_2+\cdots+(m_g-1)S_g],$$

m_i is the weight (sum of relevance score values) of the ith cluster C_i, and S_i is the covariance matrix of C_i, for $i = 1, \ldots, g$.

The classifier is based on the Bayesian classification function [9] and it uses means, covariance matrices, and weights of clusters at the cluster-merging stage of the previous iteration as prior information. The classification rule is as follows:

$$\text{Allocate } x \text{ to } C_k \text{ if } w_k f_k(x) > w_i f_i(x) \text{ for all } i \neq k \quad (8)$$

where f_i is the probability density function of C_i.

At the feedback loop stage, the user specifies a score value v for each image x. Later, after the cluster-merging stage of the current iteration, if x has become the kth point of C_i, then v becomes v_{ik}. m_i is the weight of C_i, i.e., $m_i = \sum_{k=1}^{n_i} v_{ik}$. Then w_i is the normalized weight of the ith cluster, that is, $w_i = m_i / \sum_{k=1}^{g} m_k$. The classification rule in Equation (8) is identical to one that maximizes the "posterior" probability $P(C_k \mid x) = P(x \text{ comes from } C_k \text{ given that } x \text{ was observed})$, where

$$\begin{aligned} P(C_k \mid x) &= \frac{w_k f_k(x)}{\sum_{i=1}^{g} w_i f_i(x)} \quad (9) \\ &= \frac{(prior_k) \times (likelihood_k)}{\sum[(prior_i) \times (likelihood_i)]} \text{ for } i = 1, \ldots, g \end{aligned}$$

An important special case occurs when $f_i(x)$ is a multivariate normal density function with centroid vector $\bar{x}_i$ and covariance matrix S_i of cluster C_i for $i = 1, \ldots, g$. Then Equation (8) becomes:

$$\begin{aligned} &\text{Allocate } x \text{ to } C_k \text{ if } ln(w_k f_k(x)) \geq ln(w_i f_i(x)) \text{ for all i} \neq k \\ &\text{where } ln(w_i f_i(x)) = ln(w_i) - \tfrac{p}{2} ln(2\pi) - \tfrac{1}{2} ln \mid S_i \mid \\ &-\tfrac{1}{2}(x - \bar{x}_i)' S_{pooled}^{-1}(x - \bar{x}_i) \end{aligned}$$

The constant and common terms are ignored(see Reference [9] in finding the classification function for each cluster). Then the estimate of the classification function for C_i for $i = 1, \ldots, g$ is found to be:

$$\hat{d}_i(x) = -\frac{1}{2}(x - \bar{x}_i)' S_{pooled}^{-1}(x - \bar{x}_i) + ln(w_i) \quad (10)$$

The basic idea of this classification algorithm is that a new point x should be classified to the nearest cluster. The details are as follows: For a given new point x, $\hat{d}_i(x)$ is calculated and x is assigned to kth cluster where $\hat{d}_k(x)$ is maximal. If the distance value is less than the effective radius of C_k, the point is placed to that cluster. Otherwise, it becomes the center of a new cluster. The effective radius and the distance are computed by using Equation (6). The classification algorithm is as follows:

Algorithm 2 Bayesian Classification
begin
1. **For** a new point in the relevant result set,
2. Compute $\hat{d}_1(x), \hat{d}_2(x), \ldots, \hat{d}_g(x)$ using Equation (10)
3. Determine the cluster k where $\hat{d}_k(x) = \max_{1 \leq i \leq g} \hat{d}_i(x)$
4. **If** $(x - \bar{x}_k)' S_k^{-1}(x - \bar{x}_k) < \chi^2(\alpha)$
5. place it in the cluster k
6. **Else** make it a separate cluster
7. **Endfor**
end

4.3 Cluster-Merging Stage

The clusters after the classification stage can be further merged into bigger clusters. Our basic idea of the cluster-merging stage is as follows: the initial clusters at the initial iteration include only one point in each of them. Consider two clusters at a time. If they are not significantly different, then merge them to one cluster. We repeat this until the number of clusters is reduced to a certain threshold.

Given g clusters, our cluster-merging algorithm finds candidate pairs of clusters to be merged. For efficient clustering, we determine the parameters of the merged clusters from those of existing clusters instead of those of points in existing clusters. When clusters are characterized by the mean vector, $\bar{x}_i$, covariance matrix, S_i, the number of elements in the cluster, n_i, and the weight of the cluster, m_i, we characterize a new cluster created by combining clusters i and j with the following statistics [12]:

$$\begin{aligned} m_{new} &= m_i + m_j \quad (11) \\ \bar{x}_{new} &= \frac{m_i}{m_{new}} \bar{x}_i + \frac{m_j}{m_{new}} \bar{x}_j \quad (12) \\ S_{new} &= \frac{m_i - 1}{m_{new-1}} S_i + \frac{m_j - 1}{m_{new-1}} S_j \\ &+ \frac{m_i m_j}{m_{new}(m_{new} - 1)} [(\bar{x}_i - \bar{x}_j)(\bar{x}_i - \bar{x}_j)'] \quad (13) \end{aligned}$$

Two clusters most likely to be merged should be "close" enough. The algorithm selects the next pair of clusters to be merged until the number of clusters reaches a given threshold. For this purpose, we compare their mean vectors. We infer the merge of the two clusters statistically from the closeness of two mean vectors $\bar{x}_i$ and $\bar{x}_j$. We use Hotelling's T^2 statistics [12] to test the equivalence of two mean vectors of a given pair of clusters.

For the statistical test, let us define:

- the points of ith cluster, $x_{i1}, x_{i2}, \ldots, x_{in_i}$, to be a random sample of size n_i from a population with a mean vector μ_i and a covariance matrix Σ_i.
- the points of jth cluster, $x_{j1}, x_{j2}, \ldots, x_{jn_j}$, to be a random sample of size n_j from a population with the mean vector μ_j and a covariance matrix Σ_j.
- $x_{i1}, x_{i2}, \ldots, x_{in_i}$ to be independent of $x_{j1}, x_{j2}, \ldots, x_{jn_j}$.

Especially, when n_i and n_j are small, we need the following assumptions:

- The populations of the two clusters follow multivariate normal distributions.
- The populations have the same covariance.

We use a pooled covariance to estimate the common covariance since we assume that the population covariances for the two clusters are nearly equal.

DEFINITION 3. *Hotelling's T^2 is defined by*

$$T^2(\bar{x}_i, \bar{x}_j) = \frac{m_i m_j}{m_i + m_j} (\bar{x}_i - \bar{x}_j)' S_{pooled}^{-1} (\bar{x}_i - \bar{x}_j) \quad (14)$$

where

$$\begin{aligned} S_{pooled} &= \frac{1}{m_i + m_j} (\sum_{k=1}^{n_i} v_{ik}(x_{ik} - \bar{x}_i)(x_{ik} - \bar{x}_i)' \\ &+ \sum_{k=1}^{n_j} v_{jk}(x_{jk} - \bar{x}_j)(x_{jk} - \bar{x}_j)'). \quad (15) \end{aligned}$$

The usual hypothesis to test the location difference is as follows:

$$H_0 : \mu_i = \mu_j \text{ and } H_1 : \mu_i \neq \mu_j$$

where μ_i is the unknown true center of C_i for $i = 1, \ldots, g$. If T^2 is too big which happens when $\bar{x}_i$ is "too far" from $\bar{x}_j$, then the null hypothesis H_0 is rejected. Note that $T^2 \approx \frac{p(m_i+m_j-2)}{m_i+m_j-p-1} F_{p,m_i+m_j-p-1}(\alpha)$ if H_0 is true. Here F_{p,m_i+m_j-p-1} (α) is the upper $(100(1-\alpha))$th percentile of F-distribution with p and $m_i + m_j - p - 1$ degrees of freedom. Therefore Reject H_0 if

$$T^2(\bar{x}_i, \bar{x}_j) = (\bar{x}_i - \bar{x}_j)'[(\frac{1}{m_i} + \frac{1}{m_j})S_{pooled}]^{-1}(\bar{x}_i - \bar{x}_j) > c^2 \quad (16)$$

where $c^2 = \frac{(m_i+m_j-2)p}{m_i+m_j-p-1} F_{p,m_i+m_j-p-1}(\alpha)$.
In other words, if T^2 is larger than $\frac{p(m_i+m_j-2)}{m_i+m_j-p-1}$ F_{p,m_i+m_j-p-1} (α), we conclude that the two clusters are different.
Repeat this until the number of current clusters is reduced to a given size. As α decreases, critical distance c^2 increases. That is, we can adjust the number of clusters to be merged by selecting a proper significance level α.

Algorithm 3 Cluster Merging
begin
1. Compute T^2-statistic values and c^2 values
   for all pairs of clusters
2. Sort them in ascending order and insert them into a queue
3. **While** Queue is not empty
4. Dequeue T^2 value and c^2 value for a pair of clusters i, j
5. Perform the likelihood ratio test of a pair of clusters i, j
   using Equation (16)
6. **If** $T^2 \leq c^2$, merge a pair of clusters i, j
7. **Else**
8. Increase critical distance c^2 using α
9. Add a pair of clusters i, j into a queue
10. **If** the number of current clusters $\leq$ a given size
11. stop the algorithm
12. **EndWhile**
end

The advantages of using T^2 are as follows:

- T^2 has been verified through various simulations in statistics and its theoretical properties are well known.
- Our cluster-merging method using T^2 can combine clusters of any shape. Especially, it can be well applied to elliptical clusters.
- To compute T^2, we can use the previous information from the earlier classification stage such as mean vectors, covariance matrices, etc.

DEFINITION 4. *Let $\vec{x} = (x_1, \ldots, x_p) \in \Re^p$. An algorithm is invariant under linear transformations if the statistic $U(\vec{x})$ is invariant under linear transformations, that is*

$$U(A\vec{x}) = U(\vec{x}),$$

where A is a $p \times p$ matrix with a proper inverse.

THEOREM 1. *Algorithm 2 and Algorithm 3 are invariant under linear transformations.*
Proof *It is enough to show that T^2, d^2, and $\hat{d}$ are invariant under the linear transformations. First, let us consider T^2.*

$$\begin{aligned} T^2(A\vec{x}) &= \frac{m_i m_j}{m_i + m_j}(A\bar{x}_i - A\bar{x}_j)' S_{pooled}^{-1}(A\vec{x})(A\bar{x}_i - A\bar{x}_j) \\ &= \frac{m_i m_j}{m_i + m_j}(\bar{x}_i - \bar{x}_j)' A'(AS_{pooled}A')^{-1}A(\bar{x}_i - \bar{x}_j) \\ &= \frac{m_i m_j}{m_i + m_j}(\bar{x}_i - \bar{x}_j)' A'(A')^{-1}S_{pooled}^{-1}A^{-1}A(\bar{x}_i - \bar{x}_j) \\ &= T^2(\vec{x}). \end{aligned}$$

The proofs are similar for d^2 and $\hat{d}$. □

Because of this property, the efficiency and quality of the proposed algorithms are almost the same for any linear transformations of circles, which include ellipsoids.

4.4 Dimension Reduction

A general problem of similarity retrieval in large image databases is that image/video descriptors are represented by high dimensional vectors. Since most data are from a very high dimension, the singularity of covariance is troublesome. To reduce the dimension, we use the popular principal components [9] instead of the original data.

4.4.1 Principal Component Analysis

If x is a p dimensional random vector with mean μ and covariance matrix Σ and Γ is the eigenvector of Σ, the principal component transformation is given by

$$z = (x - \mu)'\Gamma$$

where Γ is orthogonal, $\Gamma'\Sigma\Gamma = \Lambda$ is diagonal and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p \geq 0$. The strict positivity of the eigenvalues λ_i is guaranteed if Σ is positive definite. Let γ_i be ith column vector of Γ. Then $z_i = (x - \mu)'\gamma_i$ and z_i is the ith principal component of x. The variance of z_i is λ_i and the expected value of z_i is 0.

4.4.2 Sample Principal Components

Let $X = (x_1, \ldots, x_n)'$, S be the sample covariance matrix of X, G be the $p \times p$ eigenvector matrix of S and L be the eigenvalue matrix of S where x_i's are column vectors in R^p and $g_{(i)}$'s are column vectors of G. Then the sample principal component is defined by direct analogy with **4.4.1** as

$$z_{(i)} = (X - 1\bar{x}')g_{(i)}$$

where $S = GLG'$. Putting the sample principal components together we get

$$Z = (X - 1\bar{x}')G$$

G transformed one $(n \times p)$ matrix to another of the same order. L is the covariance matrix of Z.

4.4.3 Hotelling's T^2 with Principal Components

Recall that

$$T^2(\bar{x}, \bar{y}) = C(\bar{x} - \bar{y})' S_{pooled}^{-1}(\bar{x} - \bar{y})$$

where C is a constant and $\bar{x}$ and $\bar{y}$ are used in place of $\bar{x}_1$ and $\bar{x}_2$, respectively. Let $COV_{pooled}(x, y)$ denote the pooled covariance of x and y. Then

$$\begin{aligned} COV_{pooled}(G'x, G'y) &= G'S_{pooled}(G')' \\ &= G'S_{pooled}G \\ &= G'(GLG')G \\ &= L \end{aligned}$$

So,

$$
\begin{aligned}
T^2(G'\bar{x}, G'\bar{y}) &= C(G'\bar{x} - G'\bar{y})'(G'S_{pooled}G)^{-1}(G'\bar{x} - G'\bar{y}) \\
&= C(\bar{x} - \bar{y})'G(G'S_{pooled}G')^{-1}G'(\bar{x} - \bar{y})' \\
&= T^2(\bar{x}, \bar{y}) \qquad (17)
\end{aligned}
$$

By using Theorem 1, $T^2(G'\bar{x}, G'\bar{y}) = T^2(\bar{x}, \bar{y}), \hat{d}_i(G'x) = \hat{d}_i(x)$ and $d^2(G'x, G'\bar{x}_i) = d^2(x, \bar{x}_i)$ holds. Let $\bar{z}_x = G'\bar{x}$ and $\bar{z}_y = G'\bar{y}$. We have a simpler form of T^2 with principal components as follows:

$$
\begin{aligned}
T^2(G'\bar{x}, G'\bar{y}) &= C(\bar{z}_x - \bar{z}_y)'(G'S_{pooled}G)^{-1}(\bar{z}_x - \bar{z}_y) \\
&= C(\bar{z}_x - \bar{z}_y)'(G'GLG'G)^{-1}(\bar{z}_x - \bar{z}_y) \\
&= C\sum_{j=1}^{p}(\bar{z}_{xj} - \bar{z}_{yj})^2/\lambda_j \qquad (18)
\end{aligned}
$$

Note that T^2 becomes a quadratic form which saves a lot of computing efforts. Likewise, we have a simpler form of $\hat{d}_i, d^2$ with principal components.

4.4.4 Dimension Reduction in Hotelling's T^2

The proposed measures such as equations (5),(10),(14) make use of the sample covariance matrix and its inverse. To resolve the singularity problem, we adopt a new scheme using the diagonal matrix instead of the inverse covariance matrix. Let us take the first $k \leq p$ principal components such that

$$\frac{\lambda_1 + \ldots + \lambda_k}{\lambda_1 + \ldots + \lambda_k + \ldots + \lambda_p} \geq 1 - \epsilon$$

where $\epsilon \leq 0.15$. $1 - \epsilon$ is the proportion of total variation covered by the first k principal components. Let G_k be a $(p \times k)$ matrix, where columns are the first k columns of G. Let $\bar{z}_{xk} = G'_k\bar{x}$ and $\bar{z}_{yk} = G'_k\bar{y}$.

$$
\begin{aligned}
T_k^2(G'_k\bar{x}, G'_k\bar{y}) &= C(\bar{z}_{xk} - \bar{z}_{yk})'(G'_kS_{pooled}G_k)^{-1}(\bar{z}_{xk} - \bar{z}_{yk}) \\
&= C(\bar{z}_{xk} - \bar{z}_{yk})'(G'_kGLG'G_k)^{-1}(\bar{z}_{xk} - \bar{z}_{yk}) \\
&\cong C\sum_{j=1}^{k}(\bar{z}_{xkj} - \bar{z}_{ykj})^2/l_j \qquad (19)
\end{aligned}
$$

In this case, Hotelling's T^2 becomes a simple quadratic form. Likewise, we have a similar form of $d^2, \hat{d}_i$

4.5 Quality of Clustering

A good way of measuring the quality of the proposed classification method is to calculate its "classification error rates," or misclassification probabilities. Our method of measuring the clustering quality is as follows:

After the number of clusters is fixed at the final iteration, take out one element of a cluster. Check if the element is classified into the previous cluster again according to the classification procedure. Let C be the number of elements classified correctly to its own cluster and N be the total number of elements in all clusters. The error-rate becomes $1 - C/N$. This method can be applied even though numbers of elements of the cluster are small.

5. EXPERIMENTAL EVALUATION

Our extensive experiments have been conducted for two goals. First, evaluate the query clustering approach(Qcluster) to answer multipoint k-NN queries and compare it to the query point movement(QPM) and the query expansion approach(QEX). Second, test that the proposed Qcluster algorithm converges to the user's true information needs fast. We have implemented Qcluster in C++ on a Sun Ultra II. For experimental studies, the synthetic data and the Corel and Mantan image collection are used as the test set of data. The image collection includes 30,000 color images. Its images have been classified into distinct categories by domain professionals and there are about 100 images in each category. In the experiments, we use high-level category information as the ground truth to obtain the relevance feedback since the user wants to retrieve the images based on high-level concepts, not low-level feature representations [19]. That is, images from the same category are considered most relevant and images from related categories (such as flowers and plants) are considered relevant.

First, we consider the real data. In our system, we use two visual features: color moments and co-occurrence matrix texture. For color moments, we use the HSV color space because of its perceptual uniformity of color. For each of three color channels, we extract the mean, standard deviation, and skewness, and reduce the length of the feature vector to three using the principal component analysis. Then, we use the three dimensional feature vector as the color feature.

For the co-occurrence matrix texture, the (i, j)th element of co-occurrence matrix is built by counting the number of pixels, the gray-level(usually 0-255) of which is i and the gray-level of its adjacent pixel is j, in the image. Texture feature values are derived by weighting each of the co-occurrence matrix elements and then summing these weighted values to form the feature value. We extract a vector of the texture feature whose 16 elements are energy, inertia, entropy, homogentity, etc [13] and reduce the length of feature vector to four using the principal component analysis.

In the experiments, we generate 100 random initial queries and evaluate the retrieval quality for a sequence of iterations starting with these initial queries. We perform five feedback iterations in addition to the initial query. All the measurements are averaged over 100 queries. The k-NN query is used to accomplish the similarity-based match and we set k to 100. We use the hybrid tree [6] to index feature vectors of the whole data and fix the node size to 4KB.

Figure 6 compares the CPU cost of an inverse matrix scheme and a diagonal matrix scheme for the Qcluster approach when color moments are used as a feature. The diagonal matrix scheme of the Qcluster approach significantly outperforms the inverse matrix scheme in terms of CPU time. Therefore, we use a diagonal matrix scheme in our method.

Figure 7 compares the execution cost for the three approaches. The proposed Qcluster shows the similar performance with the multipoint approach [7] and outperforms the centroid-based approach such as MARS [6] and FALCON [20]. This is because our k-NN search is based on the multipoint approach that saves the execution cost of an iteration by caching the information of index nodes generated during the previous iterations of the query.

Figure 8 and 9 show the precision-recall graphs for our method when color moments and co-ocurrence matrix texture are used, respectively. In these graphs, one line is plotted per iteration. Each line is drawn with 100 points, each of which shows precision and recall as the number of retrieved images increases from 1 to 100. Based on these figures, we make two observations as follows:

- The retrieval quality improves at each iteration.
- The retrieval quality increases most at the first iteration. At the following iterations there are minor increases in the retrieval quality. This ensures that our method converges to the user's true information need fast.

Figure 10 and 11 compare the recall for query clustering, query point movement, and query expansion at each iteration. Figure 12 and 13 compare the precision for the three approaches. They produce the same precision and the same recall for the initial query. These figures show that the precision and the recall of our method increase at each iteration and outperform those of the query point movement and the query expansion approach.

Next, we performed extensive experiments to measure the accuracy of the adaptive classification algorithm and that of the cluster-merging algorithm using the synthetic data. Let $z = (z_1, \ldots, z_p)$ in $\Re^p$ where $z_1, \ldots, z_p$ are independent and identically distributed with $N(0,1)$. Then z is a multivariate normal with a mean vector 0 and a covariance matrix I, and the data shape of z is a sphere. Let $y = Az$. Then $COV(y) = AA'$ and the data shape of y is an ellipsoid. The synthetic data in $\Re^{16}$ are generated. The data consist of 3 clusters and their inter-cluster distance values vary from 0.5 to 2.5. Then the principal component analysis is used to reduce the dimension of them from 16 to 12, 9, 6, 3, respectively.

We calculate error rates of the classification algorithm ($Algorithm$ 2) with respect to 12, 9, 6, 3 dimensional data. Figure 14 shows those for spherical data and Figure 15 shows those for elliptical data when we use an inverse matrix in the Bayesian classifier. Figure 16 shows those for spherical data and Figure 17 shows those for elliptical data when we use a diagonal matrix instead. The result shows that the error rate decreases as the inter-cluster distance value increases and the error rate increases as the dimension decreases for the same inter-cluster distance value. The reason is that the information loss increases as the proportion of total variation covered by the k principal components (k=12, 9, 6, 3) decreases. Importantly, figures show that the quality of the classification algorithm stays almost the same regardless of the data shape. This result confirms the linear transformation invariant property of the proposed classification algorithm.

Next, we compute the error-ratios of the T^2-statistic with an inverse matrix and those with a diagonal matrix in order to measure the accuracy of cluster-merging algorithm with respect to 12, 9, 6, 3 dimensional data.Given 100 pairs of clusters of size 30, 100 T^2 values and corresponding critical distance (c^2) values are computed. Quantile-F values in Table 2 and 3 are the critical distance values given by the 95th percentile $F_{p,n-p}(0.05)$ where p is a dimension and n is the number of objects. If T^2 value is larger than corresponding c^2 value, reject H_0. That is, we decide that a pair of clusters must be separated. If a pair of clusters are close, then the error ratio increases in case of separating them. Table 2 and 3 show the average T^2 and the average error ratio(%) with respect to 12, 9, 6, 3 dimensional data for T^2 with an inverse matrix and T^2 with a diagonal matrix.

Figure 18 and 19 show the quantile-quantile plot of 100 T^2 values and 100 critical distance values for 50 pairs of clusters with same mean and 50 pairs of clusters with different mean. Critical distance values are calculated from random F value, which is a value from F-distribution without fixing a significance level, using the following equation:

$$randomF_{12,48} = \frac{\chi^2_{12}}{\chi^2_{48}} = \frac{\sum_{i=1}^{12} x_i^2}{\sum_{i=1}^{48} y_i^2} \qquad (20)$$

where $12, 48$ are degrees of freedom and x_i, y_i are random numbers following the Gaussian distribution of $N(0,1)$. To construct the quantile-quantile plot, we ordered T^2 values and critical distance values from smallest to largest and made the pairs of them into points of the plot. The line represents $T^2 = c^2$. The result shows that the T^2 value is less than or equal to the corresponding c^2 value when each pair of clusters have the same mean, while the T^2 value is larger than the corresponding c^2 value when each pair of clusters have different means. That is, the T^2 and the c^2 used in our cluster-merging algorithm ($Algorithm$ 3) are useful in deciding whether to merge a pair of close clusters.

6. CONCLUSION

We have focused on the problem of finding multiple clusters of a query, based on the relevance feedback, to guess the distance function and the ideal query point that the user has in mind. Our approach consists of two steps: (1) an adaptive classification that attempts to place relevant images in the current clusters or new clusters, and (2) cluster-merging that reduces the number of clusters by merging certain clusters to reduce the number of query points in the next iteration.

The major contribution in this approach is the introduction of unified quadratic forms for the distance function, the adaptive classifier, and the cluster-merging measure. Their benefit is to support the same high retrieval quality regardless of query shapes since they are linear transformation invariant in the feature space.

Our experiment shows that the proposed techniques provide a significant improvement over the query point movement and the query expansion in terms of the retrieval quality.

Table 2: Comparison of T^2 with inverse matrix and T^2 with diagonal matrix when each pair of clusters have same means

dim	variation ratio	T^2 with inverse matrix: T^2	quantile-F	error-ratio(%)
12	0.996	0.77	1.96	0
9	0.97	1.02	2.07	1
6	0.96	0.79	2.28	2
3	0.94	0.44	2.77	2

dim	variation ratio	T^2 with diagonal matrix: T^2	quantile-F	error-ratio(%)
12	0.996	0.70	1.96	2
9	0.97	0.87	2.07	4
6	0.96	0.68	2.28	6
3	0.94	0.44	2.77	6

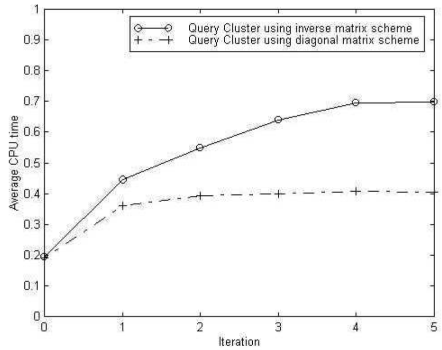

Figure 6: CPU time for inverse and diagonal matrix schemes in query cluster approach

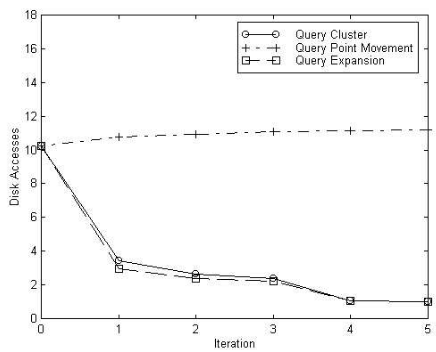

Figure 7: Comparison of execution cost for the three approaches

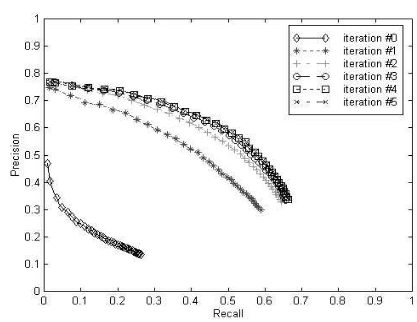

Figure 8: Precision recall graph for query clustering when color moments is used

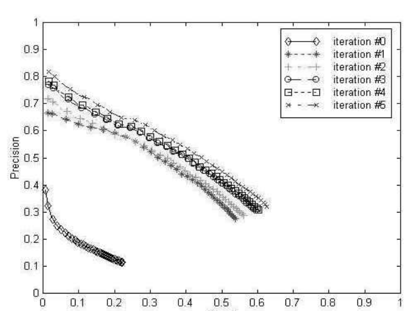

Figure 9: Precision recall graph for query clustering when co-occurrence matrix texture is used

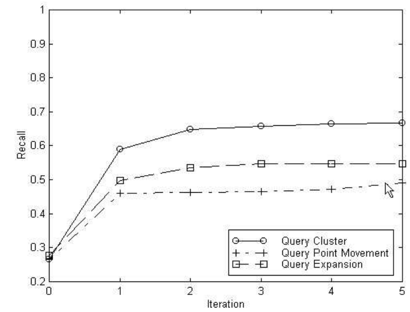

Figure 10: Comparison of recall for the three approaches when color moments is used

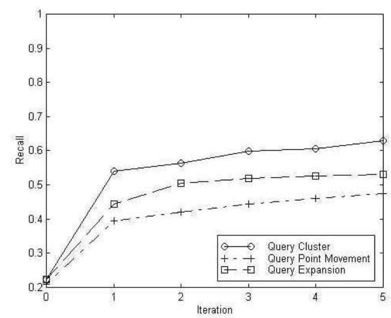

Figure 11: Comparison of recall for the three approaches when co-occurrence matrix texture is used

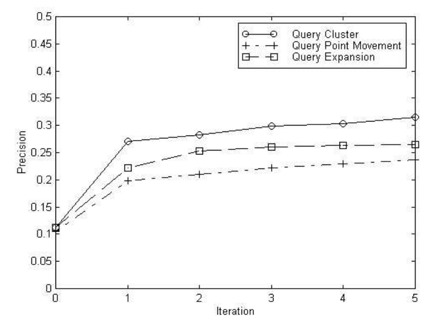

Figure 12: Comparison of precision for the three approaches when color moments is used

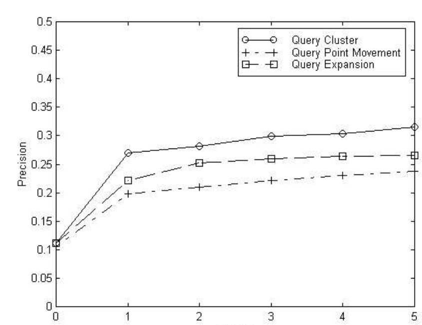

Figure 13: Comparison of precision for the three approaches when co-occurrence matrix texture is used

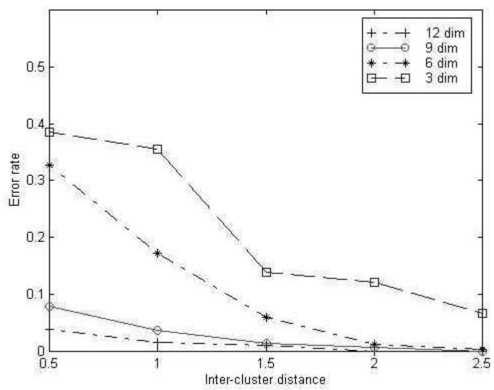

Figure 14: Error rate of the classification algorithm using an inverse matrix for 3 clusters of the spherical shape

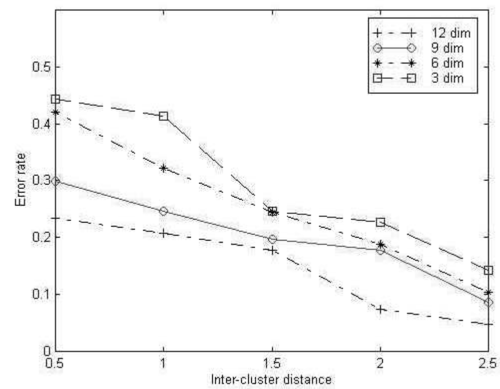

Figure 17: Error rate of the classification algorithm using a diagonal matrix for 3 clusters of the elliptical shape

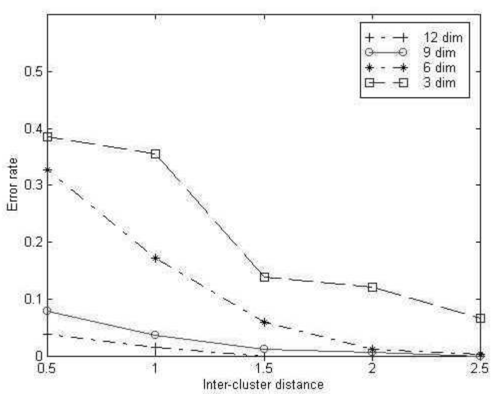

Figure 15: Error rate of the classification algorithm using an inverse matrix for 3 clusters of the elliptical shape

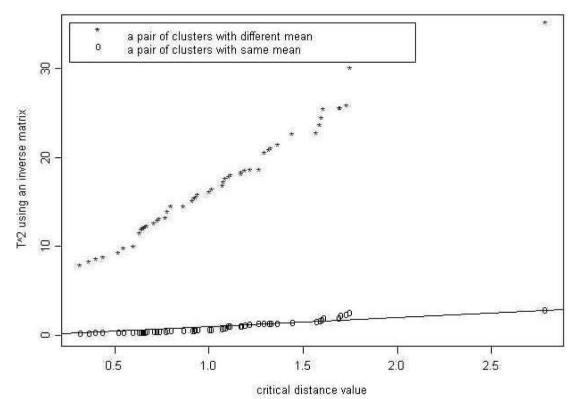

Figure 18: Q-Q plot of 100 pairs of clusters by cluster-merging algorithm using an inverse matrix

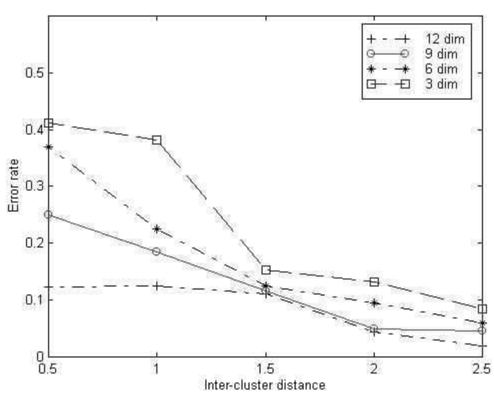

Figure 16: Error rate of the classification algorithm using a diagonal matrix for 3 clusters of the spherical shape

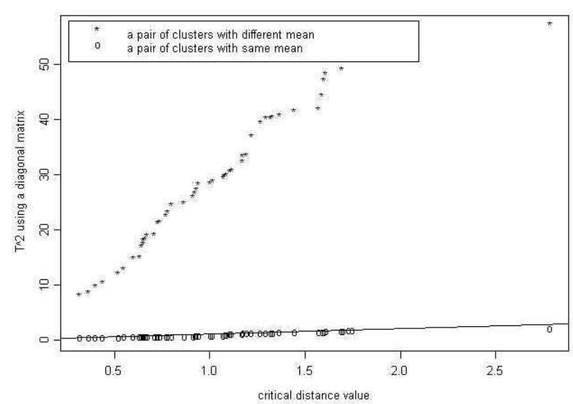

Figure 19: Q-Q plot of 100 pairs of clusters by cluster-merging algorithm using a diagonal matrix

Table 3: Comparison of T^2 with inverse matrix and T^2 with diagonal matrix when each pair of clusters have different means

dim	variation ratio	T^2 with inverse matrix		
		T^2	quantile-F	error-ratio(%)
12	0.996	20.54	1.96	0
9	0.97	24.17	2.07	0
6	0.96	31.01	2.28	0
3	0.94	38.29	2.77	6
dim	variation ratio	T^2 with diagonal matrix		
		T^2	quantile-F	error-ratio(%)
12	0.996	28.37	1.96	0
9	0.97	25.03	2.07	1
6	0.96	31.27	2.28	2
3	0.94	41.20	2.77	8

7. ACKNOWLEDGEMENTS

This work was supported by Korea Research Foundation Grant(KRF-2000-041-E00262).

8. REFERENCES

[1] T. Ashwin, R. Gupta, and S. Ghosal. Adaptable similarity search using non-relevant information. In *Proceedings of the 28th VLDB Conference*. Hong Kong, China, August 2002.

[2] I. Bartolini, P. Ciacci, and F. Waas. Feedbackbypass: A new approach to interactive similarity query processing. In *Proceedings of the 27th VLDB Conference*, pages 201–210. Roma, Italy, September 2001.

[3] M. Bauer, U. Gather, and M. Imhoff. The identification of multiple outliers in online monitoring data. Technical Report 29, SFB 475, University of Dortmund, 1999.

[4] B. Bhanu and A. Dong. Concepts learning with fuzzy clustering and relevance feedback. In *Proceedings Workshop on Machine Learning and Data Mining in Pattern Recognition*, pages 102–116, July 2001.

[5] R. Brunelli and O. Mich. Image retrieval by examples. *IEEE Transactions on Multimedia*, 2(3):164–171, September 2000.

[6] K. Chakrabarti and S. Mehrotra. The hybrid tree: An index structure for high dimensional feature spaces. In *Proceedings of 5th Int'l Conference on Data Engineering*, pages 440–447. Sydney, Australia, 1999.

[7] K. Chakrabarti, K. Porkaew, and S. Mehrotra. Efficient query refinement in multimedia databases. In *Proceedings of the IEEE Int'l Conference on Data Engineering*, page 196. San Diego, USA, Feburary 2000.

[8] M. Charikar, C. Chekuri, T. Feder, and R. Motwani. Incremental clustering and dynamic information retrieval. In *Proceedings of the ACM STOC Conference*, pages 626–635, 1997.

[9] R. D. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. John Wiley & Sons, Inc., New York, 2001.

[10] M. Flickner, H. Sawhney, and W. N. et al. Query by image and video content: The qbic system. *IEEE Computer Magazine*, 28(9):23–32, 1995.

[11] Y. Ishikawa, R. Subramanya, and C. Faloutsos. Mindreader: Querying databases through multiple examples. In *Proceedings of the 24th VLDB Conference*, pages 218–227. New York, USA, 1998.

[12] R. A. Johnson and D. W. Wichern. *Applied Multivariate Statistical Analysis*. Prentice-Hall, N.J., 1998.

[13] K. Porkaew and K. Chakrabarti. Query refinement for multimedia similarity retrieval in mars. In *Proceedings of the 7th ACM Multimedia Conference*, pages 235–238. Orlando, Florida, 1999.

[14] J. J. Rocchio. *Relevance Feedback in Information Retrieval, in G. Salton ed., The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice-Hall, Englewood Cliffs, N.J., 1971.

[15] Y. Rui, T. Huang, and S. Mehrotra. Content-based image retrieval with relevance feedback in mars. In *Proceedings of IEEE International Conference on Image Processing '97*. Santa Barbara, CA, October 1997.

[16] Y. Rui, T. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: A power tool for interactive content-based image retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(5):644–655, 1998.

[17] G. Salton, E. A. Fox, and H. Wu. Extended boolean information retrieval. *Communications of the ACM*, 26(11):1022–1036, November 1983.

[18] J. R. Smith and S. F. Chang. Visualseek: A fully automated content-based image query system. In *Proceedings of the ACM Int'l Multimedia Conference*, pages 87–98, November 1996.

[19] T. Wang, Y. Rui, and S.-M. Hu. Optimal adaptive learning for image retrieval. In *Proceedings of IEEE CVPR 2001*, pages 1140–1147. Kauai, Hawaii, 2001.

[20] L. Wu, C. Faloutsos, K. Sycara, and T. R. Payne. Falcon: Feedback adaptive loop for content-based retrieval. In *Proceedings of the 26th VLDB Conference*, pages 297–306. Cairo, Egypt, 2000.

[21] X. S. Zhou and T. S. Huang. Comparing discriminating transformations and svm for learning during multimedia retrieval. In *Proceedings of the 9th ACM Multimedia Conference*, pages 137–146. Orlando, Florida, 2001.